

# Creation of a Map through Python

Lance Duncan

Geog 375, Spring 2014

Prof. Nathan Jennings

# Purpose

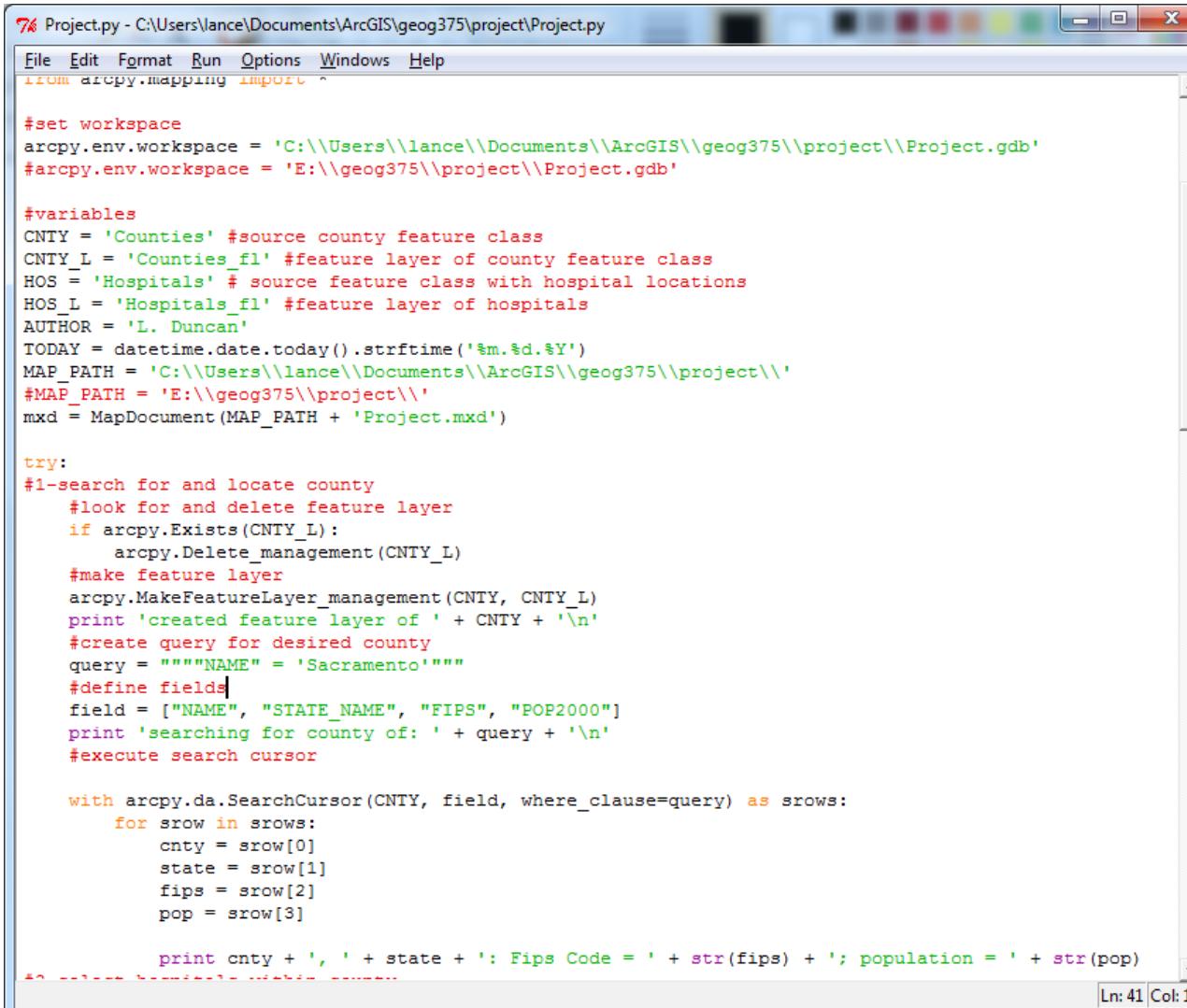
Write a script which would allow the user to input a county

Generate map showing all the hospitals in that county

# Tasks

- Roadblocks led to two different scripts
- I abandoned the first
- My second attempt proved successful

# Attempt 1



The image shows a screenshot of a Windows Notepad window. The title bar reads "Project.py - C:\Users\lance\Documents\ArcGIS\geog375\project\Project.py". The window contains Python code for an ArcGIS project. The code sets the workspace, defines variables for county and hospital feature classes, and performs a search cursor operation to find the Sacramento county. The code is color-coded for syntax.

```
76 Project.py - C:\Users\lance\Documents\ArcGIS\geog375\project\Project.py
File Edit Format Run Options Windows Help
from arcpy.mapping import *

#set workspace
arcpy.env.workspace = 'C:\\\\Users\\\\lance\\\\Documents\\\\ArcGIS\\\\geog375\\\\project\\\\Project.gdb'
#arcpy.env.workspace = 'E:\\\\geog375\\\\project\\\\Project.gdb'

#variables
CNTY = 'Counties' #source county feature class
CNTY_L = 'Counties_fl' #feature layer of county feature class
HOS = 'Hospitals' # source feature class with hospital locations
HOS_L = 'Hospitals_fl' #feature layer of hospitals
AUTHOR = 'L. Duncan'
TODAY = datetime.date.today().strftime('%m.%d.%Y')
MAP_PATH = 'C:\\\\Users\\\\lance\\\\Documents\\\\ArcGIS\\\\geog375\\\\project\\\\'
#MAP_PATH = 'E:\\\\geog375\\\\project\\\\'
mxd = MapDocument(MAP_PATH + 'Project.mxd')

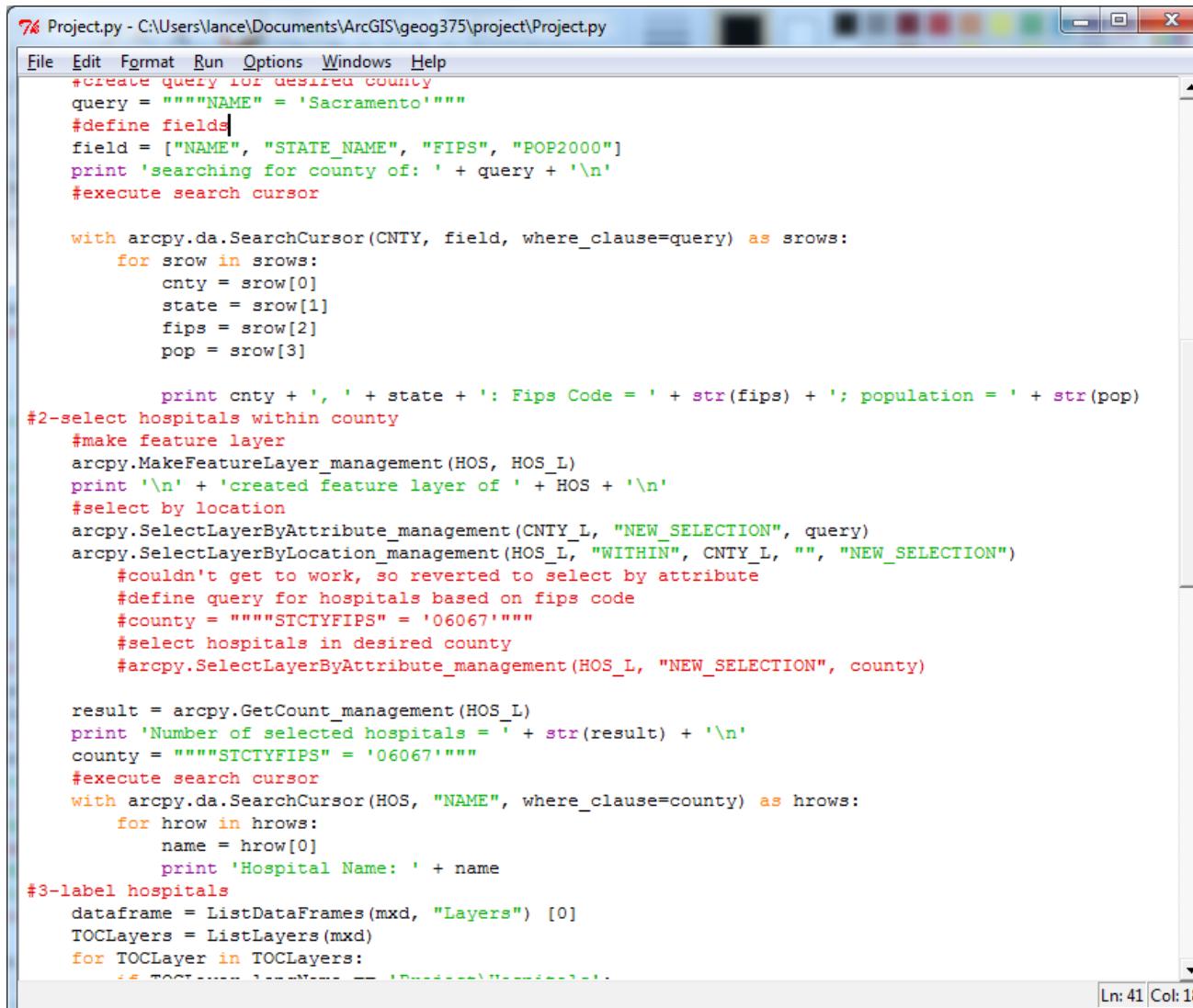
try:
    #1-search for and locate county
    #look for and delete feature layer
    if arcpy.Exists(CNTY_L):
        arcpy.Delete_management(CNTY_L)
    #make feature layer
    arcpy.MakeFeatureLayer_management(CNTY, CNTY_L)
    print 'created feature layer of ' + CNTY + '\n'
    #create query for desired county
    query = """NAME = 'Sacramento'"""
    #define fields|
    field = ["NAME", "STATE_NAME", "FIPS", "POP2000"]
    print 'searching for county of: ' + query + '\n'
    #execute search cursor

    with arcpy.da.SearchCursor(CNTY, field, where_clause=query) as srows:
        for srow in srows:
            cnty = srow[0]
            state = srow[1]
            fips = srow[2]
            pop = srow[3]

            print cnty + ', ' + state + ': Fips Code = ' + str(fips) + '; population = ' + str(pop)

```

# Attempt 1



```
76 Project.py - C:\Users\lance\Documents\ArcGIS\geog375\project\Project.py
File Edit Format Run Options Windows Help
# create query for desired county
query = """NAME" = 'Sacramento'''"
#define fields
field = ["NAME", "STATE_NAME", "FIPS", "POP2000"]
print 'searching for county of: ' + query + '\n'
#execute search cursor

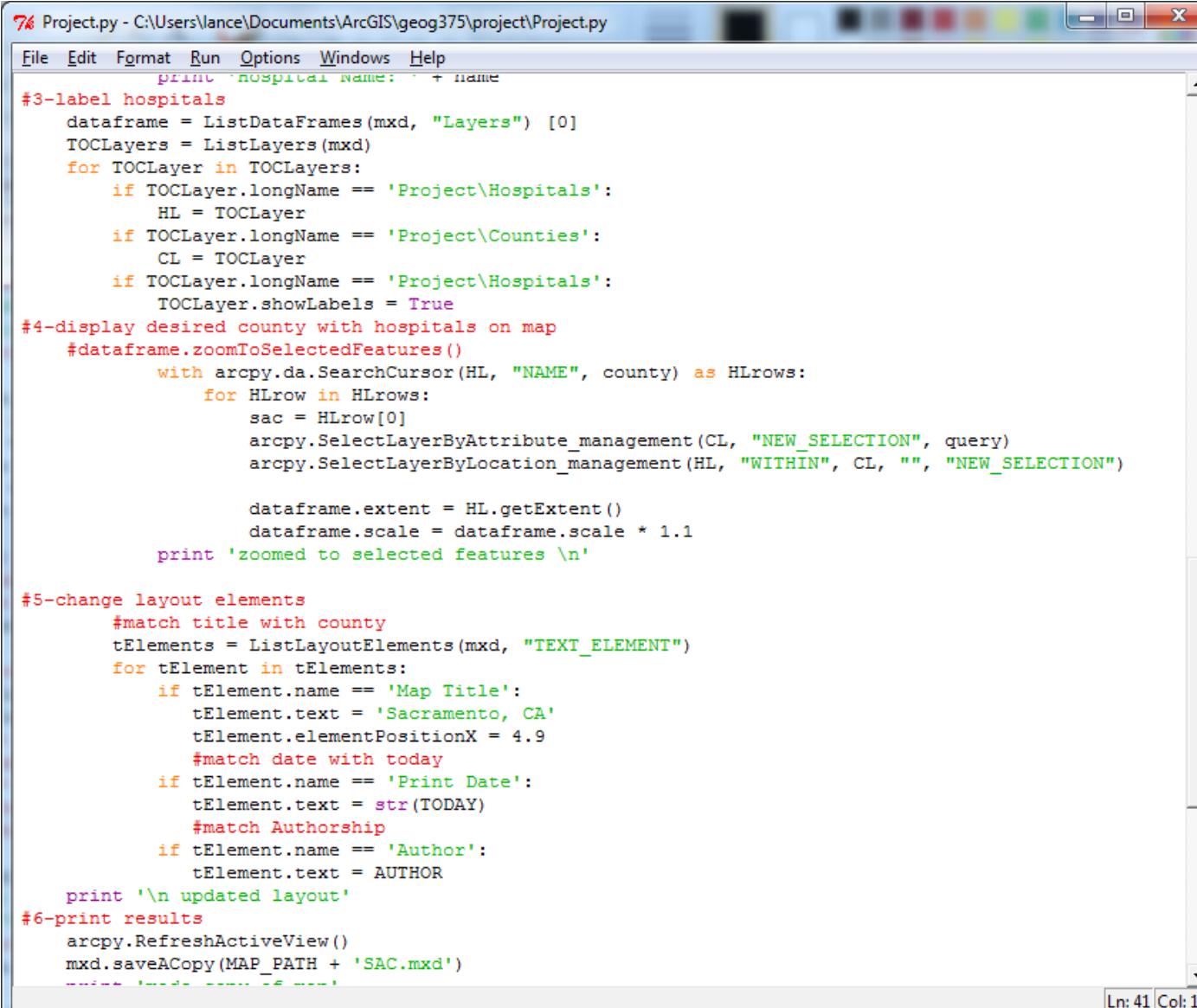
with arcpy.da.SearchCursor(CNTY, field, where_clause=query) as srows:
    for srow in srows:
        cnty = srow[0]
        state = srow[1]
        fips = srow[2]
        pop = srow[3]

        print cnty + ', ' + state + ': Fips Code = ' + str(fips) + '; population = ' + str(pop)
#2-select hospitals within county
#make feature layer
arcpy.MakeFeatureLayer_management(HOS, HOS_L)
print '\n' + 'created feature layer of ' + HOS + '\n'
#select by location
arcpy.SelectLayerByAttribute_management(CNTY_L, "NEW_SELECTION", query)
arcpy.SelectLayerByLocation_management(HOS_L, "WITHIN", CNTY_L, "", "NEW_SELECTION")
    #couldn't get to work, so reverted to select by attribute
    #define query for hospitals based on fips code
    #county = """STCTYFIPS" = '06067'''"
    #select hospitals in desired county
    #arcpy.SelectLayerByAttribute_management(HOS_L, "NEW_SELECTION", county)

    result = arcpy.GetCount_management(HOS_L)
    print 'Number of selected hospitals = ' + str(result) + '\n'
    county = """STCTYFIPS" = '06067'''"
#execute search cursor
with arcpy.da.SearchCursor(HOS, "NAME", where_clause=county) as hrows:
    for hrow in hrows:
        name = hrow[0]
        print 'Hospital Name: ' + name
#3-label hospitals
dataframe = ListDataFrames(mxd, "Layers") [0]
TOCLayers = ListLayers(mxd)
for TOCLayer in TOCLayers:
    #< TOCTITEM> . . . . .
```

Ln: 41 Col: 18

# Attempt 1



The screenshot shows a Python script titled 'Project.py' running in a code editor. The script is designed to automate the labeling and styling of a map document. It includes sections for labeling hospitals, displaying the desired county with hospitals on the map, changing layout elements to match specific details (like the map title to 'Sacramento, CA'), and printing the results. The code uses arcpy modules for interacting with the map document.

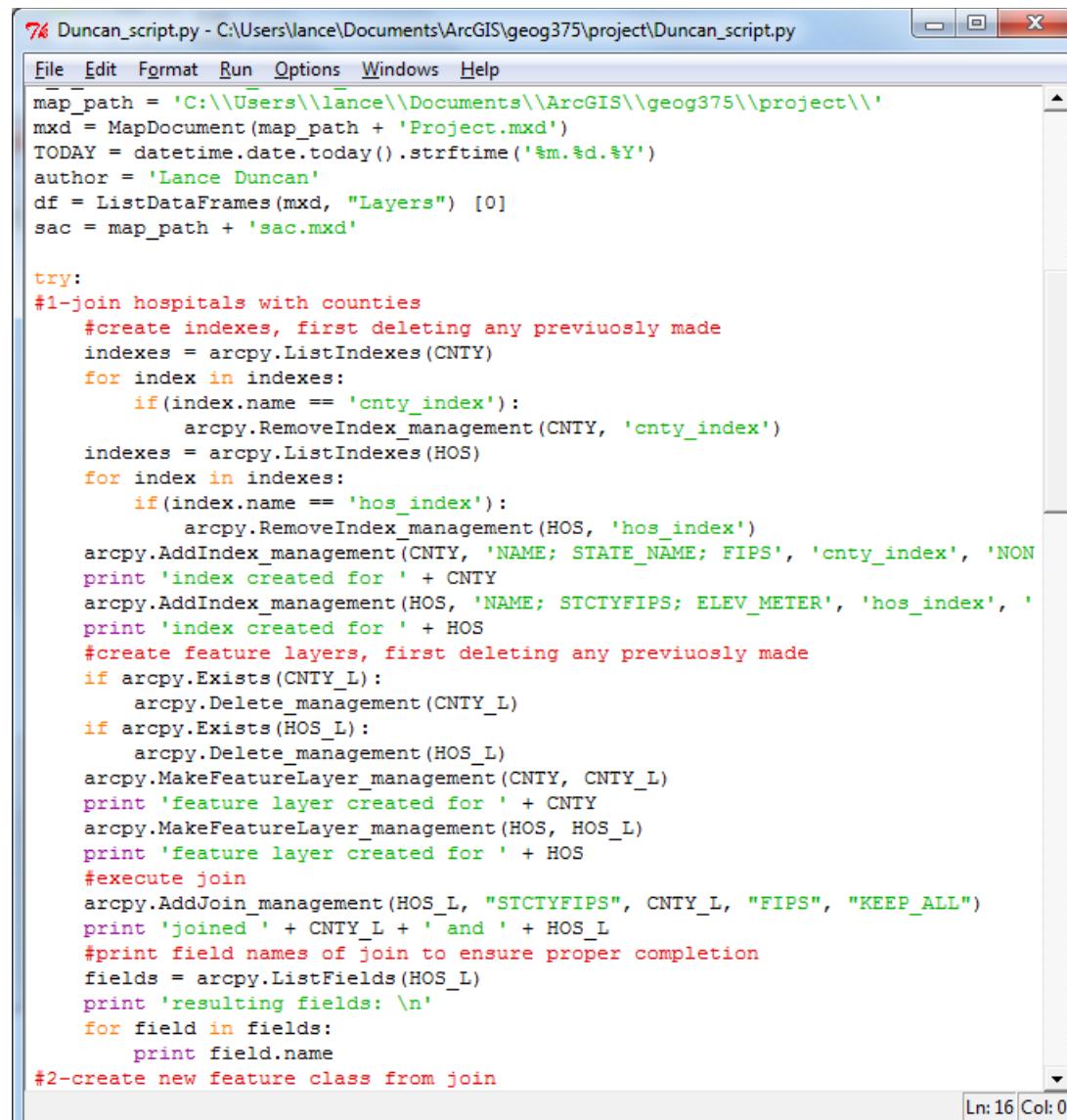
```
76 Project.py - C:\Users\lance\Documents\ArcGIS\geog375\project\Project.py
File Edit Format Run Options Windows Help
print 'hospital name: ' + name
#3-label hospitals
dataframe = ListDataFrames(mxd, "Layers") [0]
TOCLayers = ListLayers(mxd)
for TOCLayer in TOCLayers:
    if TOCLayer.longName == 'Project\Hospitals':
        HL = TOCLayer
    if TOCLayer.longName == 'Project\Counties':
        CL = TOCLayer
    if TOCLayer.longName == 'Project\Hospitals':
        TOCLayer.showLabels = True
#4-display desired county with hospitals on map
#dataframe.zoomToSelectedFeatures()
    with arcpy.da.SearchCursor(HL, "NAME", county) as HLrows:
        for HLrow in HLrows:
            sac = HLrow[0]
    arcpy.SelectLayerByAttribute_management(CL, "NEW_SELECTION", query)
    arcpy.SelectLayerByLocation_management(HL, "WITHIN", CL, "", "NEW_SELECTION")

    dataframe.extent = HL.getExtent()
    dataframe.scale = dataframe.scale * 1.1
    print 'zoomed to selected features \n'

#5-change layout elements
#match title with county
tElements = ListLayoutElements(mxd, "TEXT_ELEMENT")
for tElement in tElements:
    if tElement.name == 'Map Title':
        tElement.text = 'Sacramento, CA'
        tElement.elementPositionX = 4.9
    #match date with today
    if tElement.name == 'Print Date':
        tElement.text = str(TODAY)
    #match Authorship
    if tElement.name == 'Author':
        tElement.text = AUTHOR
    print '\n updated layout'
#6-print results
arcpy.RefreshActiveView()
mxd.saveACopy(MAP_PATH + 'SAC.mxd')
-----
```

Ln: 41 Col: 18

# Attempt 2



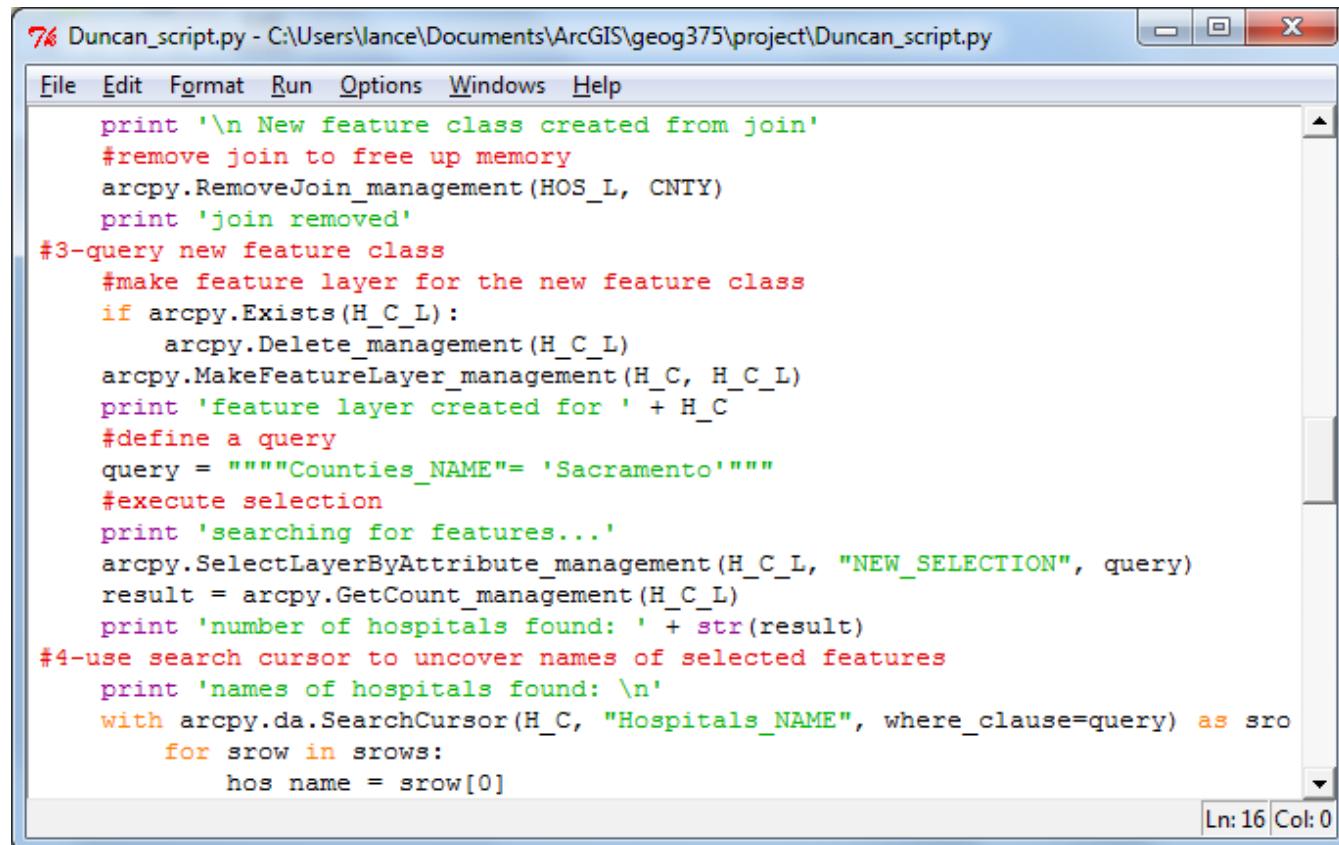
The image shows a screenshot of a Notepad window with the title bar "76 Duncan\_script.py - C:\Users\lance\Documents\ArcGIS\geog375\project\Project.mxd". The window contains a Python script for ArcGIS. The script performs the following tasks:

- Imports arcpy and sets map\_path to the project directory.
- Creates a MapDocument object (mxd) and gets the current date (TODAY).
- Creates a DataFrame object (df) for the "Layers" data frame.
- Creates a save path (sac) for the output map document.
- Attempts to join hospitals with counties, first deleting any previously made indexes for CNTY and HOS.
- Creates new indexes for CNTY and HOS using the 'NAME; STATE\_NAME; FIPS' fields.
- Prints messages for each index creation.
- Creates feature layers for CNTY and HOS, first deleting any previously made layers.
- Creates a join between HOS and CNTY using the "STCTYFIPS" field.
- Prints the joined layers and lists the resulting fields.
- Creates a new feature class from the joined data.

```
76 Duncan_script.py - C:\Users\lance\Documents\ArcGIS\geog375\project\Project.mxd
File Edit Format Run Options Windows Help
map_path = 'C:\\\\Users\\\\lance\\\\Documents\\\\ArcGIS\\\\geog375\\\\project\\\\'
mxd = MapDocument(map_path + 'Project.mxd')
TODAY = datetime.date.today().strftime('%m.%d.%Y')
author = 'Lance Duncan'
df = ListDataFrames(mxd, "Layers") [0]
sac = map_path + 'sac.mxd'

try:
#1-join hospitals with counties
    #create indexes, first deleting any previuosly made
    indexes = arcpy.ListIndexes(CNTY)
    for index in indexes:
        if(index.name == 'cnty_index'):
            arcpy.RemoveIndex_management(CNTY, 'cnty_index')
    indexes = arcpy.ListIndexes(HOS)
    for index in indexes:
        if(index.name == 'hos_index'):
            arcpy.RemoveIndex_management(HOS, 'hos_index')
    arcpy.AddIndex_management(CNTY, 'NAME; STATE_NAME; FIPS', 'cnty_index', 'NON
print 'index created for ' + CNTY
    arcpy.AddIndex_management(HOS, 'NAME; STCTYFIPS; ELEV_METER', 'hos_index', 'NON
print 'index created for ' + HOS
#create feature layers, first deleting any previuosly made
if arcpy.Exists(CNTY_L):
    arcpy.Delete_management(CNTY_L)
if arcpy.Exists(HOS_L):
    arcpy.Delete_management(HOS_L)
arcpy.MakeFeatureLayer_management(CNTY, CNTY_L)
print 'feature layer created for ' + CNTY
    arcpy.MakeFeatureLayer_management(HOS, HOS_L)
print 'feature layer created for ' + HOS
#execute join
    arcpy.AddJoin_management(HOS_L, "STCTYFIPS", CNTY_L, "FIPS", "KEEP_ALL")
print 'joined' + CNTY_L + ' and ' + HOS_L
#print field names of join to ensure proper completion
fields = arcpy.ListFields(HOS_L)
print 'resulting fields: \n'
for field in fields:
    print field.name
#2-create new feature class from join
```

# Attempt 2



The screenshot shows a Windows-style application window titled "Duncan\_script.py - C:\Users\lance\Documents\ArcGIS\geog375\project\". The window contains a menu bar with File, Edit, Format, Run, Options, Windows, and Help. The main area of the window displays a Python script with syntax highlighting for ArcPy commands. The script performs the following steps:

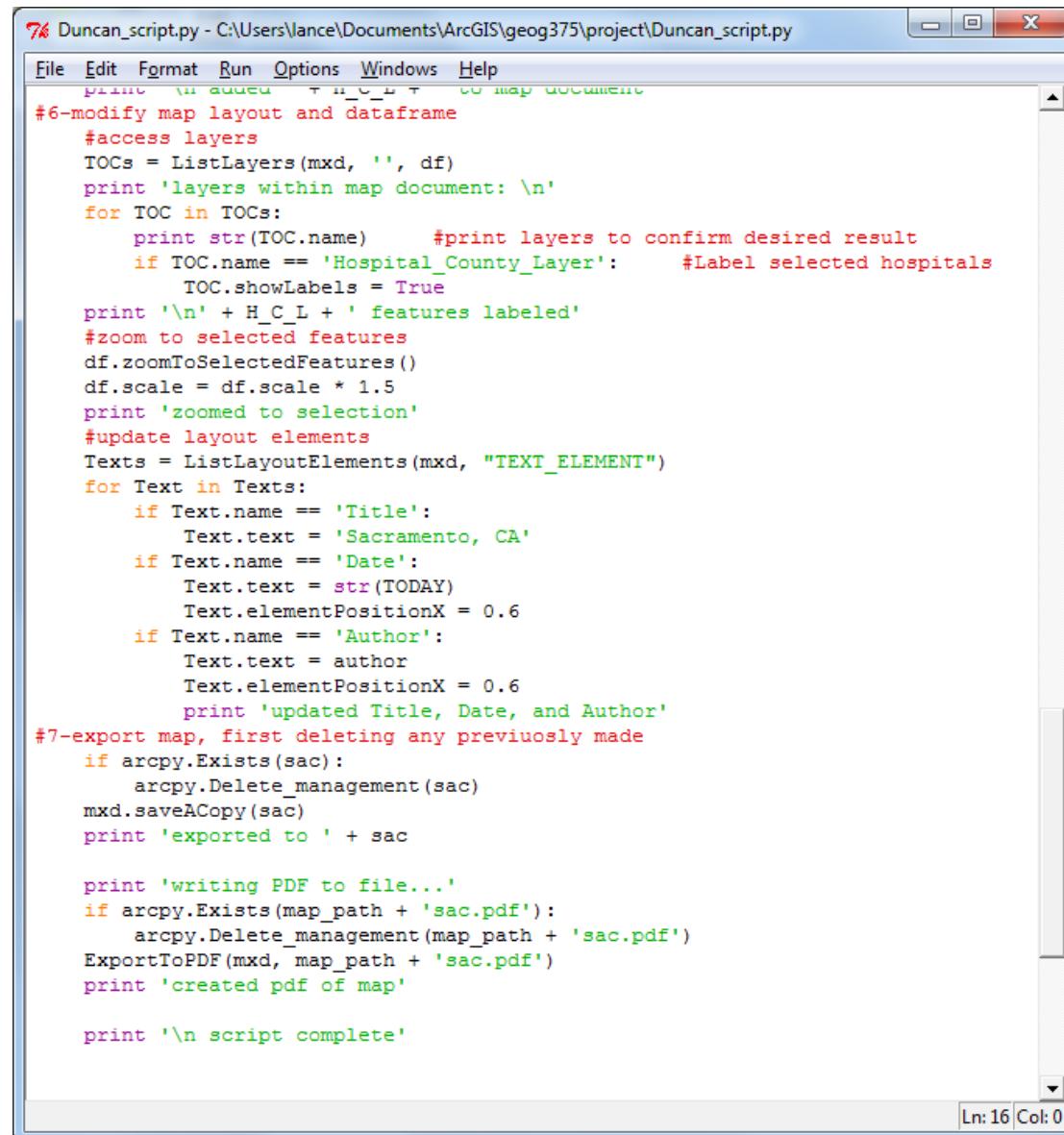
- Prints a message indicating a new feature class is created from join.
- Removes the join to free up memory using  `arcpy.RemoveJoin_management(HOS_L, CNTY)`.
- Prints a message indicating the join is removed.
- Creates a new feature class (H\_C\_L) if it does not exist, using  `arcpy.Delete_management(H_C_L)` and  `arcpy.MakeFeatureLayer_management(H_C, H_C_L)`.
- Prints a message indicating the feature layer is created for the hospital layer.
- Defines a query for the "Sacramento" county using `query = "Counties_NAME='Sacramento'"`.
- Executes the selection using  `arcpy.SelectLayerByAttribute_management(H_C_L, "NEW_SELECTION", query)`.
- Gets the count of selected features using `result = arcpy.GetCount_management(H_C_L)`.
- Prints the number of hospitals found.
- Uses a search cursor to uncover names of selected features, printing the names of hospitals found.
- Creates a search cursor for the hospital layer using `with arcpy.da.SearchCursor(H_C, "Hospitals_NAME", where_clause=query) as sro`.
- Iterates through the search cursor rows using `for srow in srows:`.
- Prints the hospital name from the first row using `hos name = srow[0]`.

The status bar at the bottom right of the window shows "Ln: 16 Col: 0".

```
76 Duncan_script.py - C:\Users\lance\Documents\ArcGIS\geog375\project\

File Edit Format Run Options Windows Help
print '\n New feature class created from join'
#remove join to free up memory
arcpy.RemoveJoin_management(HOS_L, CNTY)
print 'join removed'
#3-query new feature class
#make feature layer for the new feature class
if arcpy.Exists(H_C_L):
    arcpy.Delete_management(H_C_L)
arcpy.MakeFeatureLayer_management(H_C, H_C_L)
print 'feature layer created for ' + H_C
#define a query
query = "Counties_NAME='Sacramento'"
#execute selection
print 'searching for features...'
arcpy.SelectLayerByAttribute_management(H_C_L, "NEW_SELECTION", query)
result = arcpy.GetCount_management(H_C_L)
print 'number of hospitals found: ' + str(result)
#4-use search cursor to uncover names of selected features
print 'names of hospitals found: \n'
with arcpy.da.SearchCursor(H_C, "Hospitals_NAME", where_clause=query) as sro
    for srow in srows:
        hos name = srow[0]
Ln: 16 Col: 0
```

# Attempt 2



```
74 Duncan_script.py - C:\Users\lance\Documents\ArcGIS\geog375\project\

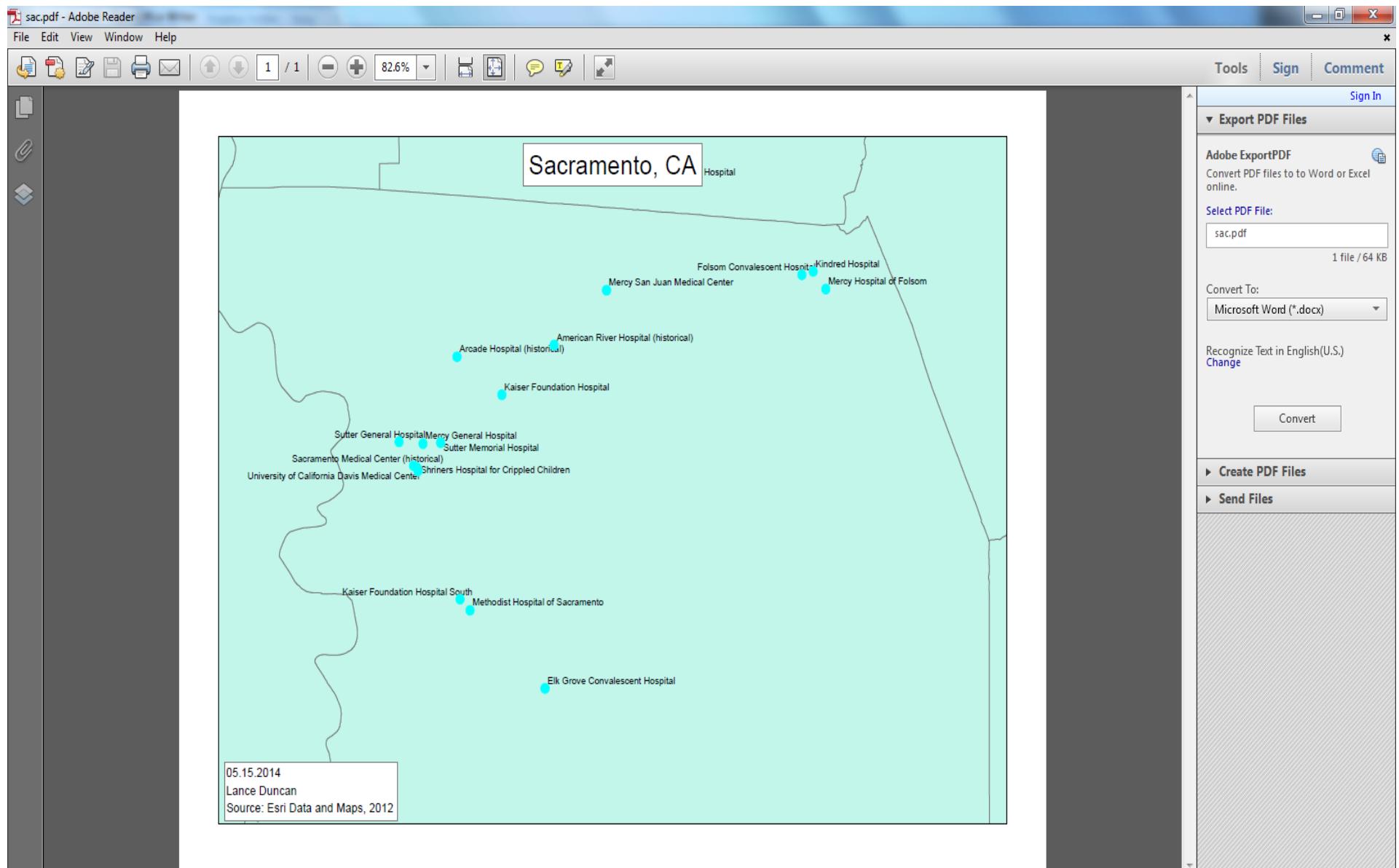
File Edit Format Run Options Windows Help
print 'in' + H_C_L + ' map document'
#6-modify map layout and dataframe
#access layers
TOCs = ListLayers(mxd, '', df)
print 'layers within map document: \n'
for TOC in TOCs:
    print str(TOC.name)      #print layers to confirm desired result
    if TOC.name == 'Hospital_County_Layer':      #Label selected hospitals
        TOC.showLabels = True
print '\n' + H_C_L + ' features labeled'
#zoom to selected features
df.zoomToSelectedFeatures()
df.scale = df.scale * 1.5
print 'zoomed to selection'
#update layout elements
Texts = ListLayoutElements(mxd, "TEXT_ELEMENT")
for Text in Texts:
    if Text.name == 'Title':
        Text.text = 'Sacramento, CA'
    if Text.name == 'Date':
        Text.text = str(TODAY)
        Text.elementPositionX = 0.6
    if Text.name == 'Author':
        Text.text = author
        Text.elementPositionX = 0.6
    print 'updated Title, Date, and Author'
#7-export map, first deleting any previuosly made
if arcpy.Exists(sac):
    arcpy.Delete_management(sac)
mxd.saveACopy(sac)
print 'exported to ' + sac

print 'writing PDF to file...'
if arcpy.Exists(map_path + 'sac.pdf'):
    arcpy.Delete_management(map_path + 'sac.pdf')
ExportToPDF(mxd, map_path + 'sac.pdf')
print 'created pdf of map'

print '\n script complete'
```

Ln: 16 Col: 0

# Result



# Conclusions

- First or most obvious approach might not be the best approach
- I was not proficient at using the mapping module
- GUI possibilities